Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

1997-09

Prototype for enhancement of ANVIS/HUD CBT instruction through use of embedded visual simulation

Foggin, G. Thomas

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/8870

Downloaded from NPS Archive: Calhoun

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

NPS ARCHIVE
1997.09
FOGGIN, G.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

PROTOTYPE FOR ENHANCEMENT OF ANVIS/HUD CBT INSTRUCTION THROUGH USE OF EMBEDDED VISUAL SIMULATION

by

G. Thomas Foggin, IV
Paul J. O'Rourke

September, 1997

Advisor: Anthony Ciavarelli
Associate Advisor: Tung Bui

Approved for public release; distribution unlimited.

Thesis
F5296

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

Approved for public release; distribution is unlimited.**REPORT DOCUMENTATION PAGE**

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1997	Master's Thesis

4. TITLE AND SUBTITLE
PROTOTYPE FOR ENHANCEMENT OF ANVIS/HUD CBT INSTRUCTION THROUGH USE OF EMBEDDED VISUAL SIMULATION

5. FUNDING NUMBERS

6. AUTHOR(S) Foggin, IV, G. Thomas, and O'Rourke, Paul J.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
NAVAIRSYSCOM, PMA-205; 1421 Jefferson Hwy; Arlington, VA. 22243

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The purpose of this project is to develop a computer based trainer (CBT) for ANVIS/HUD that takes advantage of recent advances in multimedia technology. Integration of a head mounted display (HMD) into the CBT system allows the user to be immersed into a virtual world that simulates actual NVG use. In accordance with guidelines established by Ciavarelli, Baer and Sengupta, in their NVG Training Technology Report, December 1994, for the Naval Aviation Systems Command (PMA 205) and using Macromedia Director 6.0, it is possible to incorporate a synthesized continuous multimedia data base into a system that permits user interaction along a scripted NVG flight path. The system has the capability of demonstrating some of the capabilities and limitations of an actual ANVIS/HUD system under user selectable lighting and terrain features. By utilizing commercial off the shelf (COTS) software and hardware the system represents a possible low cost, personal computer (PC) based, ANVIS/HUD trainer.

14. SUBJECT TERMS Multimedia Computer-Based Training, Embedded Simulation, Night Vision Training			15. NUMBER OF PAGES 94
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

Approved for public release; distribution is unlimited.

PROTOTYPE FOR ENHANCEMENT OF ANVIS/HUD CBT INSTRUCTION THROUGH USE OF EMBEDDED VISUAL SIMULATION

G. Thomas Foggin, IV
Lieutenant, United States Navy
B.S., United States Naval Academy, 1989

Paul J. O'Rourke
Lieutenant, United States Navy
B.S., Rochester Institute of Technology, 1989

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 1997

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

ABSTRACT

The purpose of this project is to develop a computer based trainer (CBT) for ANVIS/HUD that takes advantage of recent advances in multimedia technology. Integration of a head mounted display (HMD) into the CBT system allows the user to be immersed into a virtual world that simulates actual NVG use. In accordance with guidelines established by Ciavarelli, Baer and Sengupta, in their NVG Training Technology Report, December 1994, for the Naval Aviation Systems Command (PMA 205) and using Macromedia Director 6.0, it is possible to incorporate a synthesized continuous multimedia data base into a system that permits user interaction along a scripted NVG flight path. The system has the capability of demonstrating some of the capabilities and limitations of an actual ANVIS/HUD system under user selectable lighting and terrain features. By utilizing commercial off the shelf (COTS) software and hardware the system represents a possible low cost, personal computer (PC) based, ANVIS/HUD trainer.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. PURPOSE 1

B. BACKGROUND 1

C. PROBLEM STATEMENT 2

D. GENERAL APPROACH 3

E. RESEARCH OUTPUT 5

F. SCOPE AND LIMITATIONS 5

II. BACKGROUND 7

A. MULTIMEDIA 7

B. COMPUTER–BASED TRAINING 9

III. METHODOLOGY 13

A. OBJECTIVE 13

B. TRAINING / FUNCTIONAL REQUIREMENTS AND SPECIFICATIONS 13
1. Training Requirement 13
2. Functional Specifications 14

C. HARDWARE AND SOFTWARE DESIGN 15
1. Hardware 15
2. Software 15

D. SYSTEM DEVELOPMENT PROCEDURES 16
1. Creation of Background Imagery 16
2. Movie Generation 17

IV. RESULTS AND DISCUSSION 21

A. OVERVIEW OF VESD TRAINER 21

B. OPERATION OF THE DEMO 25

C. LESSONS LEARNED 29

V. CONCLUSIONS AND RECOMMENDATIONS 31

A. CONCLUSIONS 31

B. RECOMMENDATIONS 32

REFERENCES 35

APPENDIX A. FORTE VFX™ SPECIFICATIONS AND REQUIREMENTS 37

APPENDIX B. LINGO SCRIPT 41

APPENDIX C. INSTRUCTIONS FOR SETUP OF DEMONSTRATION SYSTEM 79

INITIAL DISTRIBUTION LIST 83

LIST OF FIGURES

4.1 Main Menu of Flight Options 22

4.2 Desert Bright Scene 23

4.3 Mountain Ridge Bright Scene 24

4.4 HUD Tutor Symbology Display Screen 25

4.5 HUD Symbology Menu 27

4.6 Instructions for Interactive Flight 28

I. INTRODUCTION

A. PURPOSE

This thesis project is part of ongoing research and development sponsored by NAVAIR (PMA– 205). Its purpose of which was to develop and test advanced training systems for naval aircrews. The primary aim of this thesis was to design and develop a working prototype demonstration system for teaching aviators how to effectively and safely use a new night vision aiding system, the Aviator's Night Vision Imaging System/Heads Up Display (ANVIS/HUD).

B. BACKGROUND

As previously stated by Kern and Shaffer, 1996:

> The recent downsizing within the Department of Defense has forced most fleet aviation components to operate with minimum resources. A prevailing working doctrine stating that we "must do more with less" dictates that aviation commands must meet readiness goals as efficiently as possible.
>
> The Navy, along with the other services, has often relied on advanced technology to improve efficiency and force effectiveness. An underlying assumption is that our forces can use technology to their benefit in attaining readiness goals and improving mission performance. However, while technological advances may provide a potentially greater tactical advantage for the pilot, they also impose on the pilot a greater burden for understanding a wide variety of equipment and theory. In short, today's aircrews are under more pressure than ever to understand the constantly changing number of assets that are available to them, hence the training systems that are delivered to the fleet must be efficient, cost effective, and as high a quality as the aircrews who use them.
>
> Training aviators has become a highly visible task, and when this training involves night operations, the training procedures are scrutinized even more. In particular, the historical costs of neglecting night vision training, in terms of both personnel and material losses, demand more effective training methods. Fleet components may not always have the manpower to properly train all aircrew on night vision equipment. This being the case, either the quality of training suffers – and safety is sacrificed – or a better way to train is developed. Computer–based

applications are quickly becoming part of the answer to many of the fleet's training problems because (1) these applications provide a means to standardize training, and (2) They deliver needed training a the unit level.

A number of NPS theses have recently been written on the development and use of multimedia trainers for use with Night Vision Goggle (NVG's) and Heads Up Displays (HUD's). In 1994 Epperson and Meza developed a system that incorporated the opportunity for the user to open windows to view video. In 1996 Kern and Shaffer developed the HH–60H ANVIS/HUD baseline training system. Their trainer did not incorporate multimedia features. Later in 1996 and early 1997 Price and Foster expanded on Kern and Shaffer's work to introduce an improved user interface. This prototype also included windows for video playback. All of these previous CBT systems had in common an SVGA computer monitor as the primary output device to the user. The output imagery itself was primarily a mix of graphics, photography and text with minimal opportunity for interaction. The primary purpose of our project was to explore the practicality of adding interactive video displayed in a Head Mounted Display (HMD), also known as a Virtual Reality helmet, to improve the realism and value of the training provided. The project was limited to relatively low cost options for practical reasons and because the resource constraints were consistent with the current fiscal reality of fleet budgets.

C. PROBLEM STATEMENT

Further multimedia enhancements to the ANVIS/HUD trainer were desired that would allow trainees to visualize the use of the ANVIS/HUD under more realistic flight conditions.

D. GENERAL APPROACH

In the earlier work on NVG training by Ciavarelli, Sengupta and Baer (1994) the elements of a prospective multimedia NVG/HUD trainer were presented (see Chapter III). Using the guidelines developed in that document we attempted to produce a proof–of–concept system that would provide simulated visual scenes viewed through an HMD using currently existing technologies. The knowledge gained building the system could be used to determine the capability of the technology and provide some insight as to what may be possible in the near future. The intent was to use commercial off the shelf (COTS) products as much as possible to develop a small scale visual simulation that could be displayed to the trainee with superimposed HUD symbology.

The project was broken up into essentially three parts. An interactive multimedia authoring environment, terrain scene development, hardware and interactivity considerations for the HMD.

A number of commercially available multimedia authoring systems were available. The one chosen, Macromedia's Director 6.0, appears to be one of the most capable on the market. It has an environment in which the author can create an interactive movie. Features include the ability to remove, add or move images around the stage based on external inputs including keyboard, mouse, joystick and voice. Additional capabilities are being added regularly via second source vendors who write pluggins, called Xtras, for Director. From Director movie playback and screen refresh speeds can be controlled. Graphical user interfaces are easily created and then displayed by replaying the same frame over and over until the user chooses to leave the frame for a movie, still picture or another screen.

For terrain development Vistapro 4.0 was chosen. Other scene generation software packages were available but none did exactly the things Vistapro can do. Vistapro allows the user to work on real terrain by using U.S. Geological Survey data and converting it into a virtual world. The scenery can then be populated with a number of terrain features (grass, trees, water etc) and colored in any way the user likes. Vistapro also provides vehicle models, in this case the helicopter model was used, to develop a movie which depicts the imagery along a user described path through the terrain. The movie can be exported in a variety of formats for ease of use.

In addition to a Personal Computer (PC) the HMD is the only one of these three elements that the user would need to acquire to take advantage of a developed ANVIS/HUD training program. The selection of the HMD was more difficult. There are basically three classes of HMD's low, medium and high cost/capability. The low cost/capability ones were ruled out as inadequate for the task. The high cost ones were also ruled out because they were too expensive. Hardware costs tend to come down so their capabilities may be more affordable in the near future. The problem with the medium cost/capability units is that they have a very limited market – namely the serious gamer. The two units we looked at were both manufactured by companies in serious financial trouble. One was actually out of business and the other in Chapter 11. The Forte VFX1 Headgear unit was chosen for several reasons. It incorporated magnetic head tracking, joystick command emulation for the tracking unit, reasonable resolution (640 X 480 pixels), headphones for sound, an independent three–dimensional mouselike input device and optical focusing characteristics (inter–pupilary distance etc.) similar to those found on NVG's. Using the VFX1 is very much like using NVG's mounted on a

pilot's helmet except for the reduced optical resolution. The unit also supports line sequential stereo mode, which the authors determined could be used to present monocular images in a binocular setting.

All of the above hardware and software components were used together to produce a system similar to the one envisioned by Ciavarelli et al in their description of a low cost, PC based, multimedia CBT for NVG training. A more detailed description of system components and procedures used during engineering development will be discussed later in Chapter III.

E. RESEARCH OUTPUT

The primary goal of this research was to develop a functional framework for developing a multimedia computer–based ANVIS/HUD trainer. The mission was to see if it was possible to build a system using COTS products and a helmet mounted display that would provide the fidelity necessary to simulate use of the ANVIS/HUD in a night environment. The functional prototype resides in the Naval Aviation Safety School Multimedia Lab, Naval Postgraduate School, Monterey, California.

This prototype is a component of an ongoing research and development project and should be viewed and operated in that regard.

F. SCOPE AND LIMITATIONS

The intent was to demonstrate what level of functionality was achievable with low cost PC equipment, an off the shelf HMD and commercially available software development tools. It should be noted that we did not set out to develop a full spectrum multimedia trainer for fleet use but rather to develop a technology demonstration system

that could be incorporated into a multimedia curriculum that would add significant training value to the baseline ANVIS/HUD trainer.

It was determined that the system should run on an IBM PC based system using the Windows 95/NT operating system since that is what the fleet currently has or was in the process of procuring (IT–21). Due to interoperability of existing systems it could possibly also be run or developed on other systems (i.e. Apple computers or SUN workstations). Using a workstation to run a trainer would have several advantages including higher speeds, available 3D game engines, etc. A number of drawbacks are inherent in using these workstations. Fleet users are not generally familiar with the UNIX operating system. Most fleet software applications are not available for UNIX machines limiting the other uses the machine might have in a small unit with limited resources. The higher cost of the systems makes them less attractive to the fleet.

The developed PC–based trainer demonstrates the interactivity of the HUD symbology with the visual environment. With relatively little training a developer could learn to create new lessons fairly quickly. It was estimated that once a scenario was established, a "movie" could be developed in less than one week.

II. BACKGROUND

A. MULTIMEDIA

Multimedia is the computer controlled integration of text, graphics, still and moving images, animation, sounds, and any other medium where every type of information can be represented, stored, transmitted, and processed digitally. The interface in these systems may permit interactivity to the final user. An interactive multimedia system is an integrated computer controlled environment that allows for the manipulation of digital information. Interactivity is the amount of control the user has over the display and presentation of information. Kirkley (1995) describes the three most common classifications of interactivity:

- A linear presentation is one in which the author decides the sequence and manner in which information is presented. The user controls only the pace.
- A programmed branching program is one in which the user has some control over the sequence of presentation by selecting from a group of choices such as from a main menu. The author still maintains the control of deciding what to include in the choices available at various points in the program.
- Hypermedia can be thought of as a web of interrelated information in which the user is in almost complete control of the pace, sequence and content of the presentation. Links provide a means for random access of information.

How it is generated and their display types classify media. The two means of media generation are synthesized and captured. Computers create synthesized media while captured media comes from the real world and is captured digitally by some

recording device. There are two general multimedia display types, discrete and continuous. Discrete media is space–based. A slide show would be an example of discrete media. Continuous media is both space and time based. Animation and motion video is examples of continuous media. (Xie, 1997)

Multimedia can be combined with computer–assited instruction to make learning highly individualized and interactive (Oblinger, 1992). Interactivity increases the learner's engagement with the learning situation (ED, 1991). These two traits are what make desk–top computer–based training so attractive.

The dynamic nature of multimedia involves many challenges and design trade offs. The main elements of multimedia are text, graphics, images, animation, full motion video, and sound. Each of the media types can be represented in various digital forms that include numerous compression schemes. It is in this arena that the developer needs to determine the balance between media types and user performance requirements. A permanent storage medium is used to keep the digitized information for future retrieval and use. Multimedia requires extensive amounts of storage during both the development and application stages. Bandwidth can be thought of as the pipe through which information flows. The larger the amount of data that needs to be delivered, the bigger the pipe needed. Bandwidth and storage capacity are currently the biggest limitations to delivering large amounts of digital video in multimedia (Kirkley, 1995). When designing a CBT, these performance and storage trade offs directly impact the fidelity and training value of the end product. Incorrect design choices will negatively impact the final product. The authors' prototype is primarily a visual trainer, therefore high fidelity imagery was used. The penalty for this design decision is very large storage

requirements and video run times of only thirty seconds at fifteen frames per second for the non–interactive portions. The interactive video storage requirements reduced run times to eight seconds in length.

Multimedia systems require high speed processing. This is due to the execution speed and large file size that sound and video data demand. In terms of bytes of information, sound and video files are the two largest media types. The current Reduced Instruction Set Computer (RISC) based processors for personal computers provide execution speeds up to 266 megahertz. Moore's law states that processing speed and storage will roughly double every 18 to 24 months and has held true for the last thirty–five years. This trend can be expected to continue. Therefore desktop multimedia systems will increase in speed, power, and fidelity.

The display depth is the systems ability to display an accurate image. Newer computers can display high fidelity images using millions of colors and are approaching photo realistic images. A recent summary of multimedia assessment methods and discussion of technology issues are presented in Nixon's (1997) thesis. Blending the requirements of computing power, data storage and management, human interface usability, latency, and throughput are difficult and necessitate a deep understanding of what the system is to present.

B. COMPUTER–BASED TRAINING

Computer–based training (CBT) uses a computer and software to teach a skill or enhance a student's knowledge base. The CBT environment may be enhanced through the use of multimedia. These interactive programs allow the user to progress through learning objectives at their own pace, and revisit those that are in question. The modules

may include theory, exercises, demonstrations, and exams that offer immediate feed back to the user. Many of these trainers provide a student–tracking feature so that an instructor can monitor a student's progress and supplements instruction when required. Low–end programs are largely text based while higher–end programs take advantage of many multimedia features.

Orlanski (1994) gives a brief summary of a few major substantive finding of interest to military training:

- Computer–based instruction saves about 30 percent of the time students need to complete a course, compared to conventional instruction.
- Interactive videodisc instruction raised performance of 50th percentile students to about the 70th percentile achievement level.
- The average cost of interactive videodisc instruction is about 40 percent that of conventional instruction.

CBT systems are typically dominated by tutorial presentations that have been limited to use of text, graphics, and photographic images. With the tremendous improvement in microprocessor area, it may now be possible to incorporate different media such as video, and interactive simulations. Such enhancements may be particularly useful in depicting portions of the operational world that would improve the training value of a CBT.

Studies by IBM and others indicate that people forget most of what they hear, retain only 20 percent of what they see, 40 percent of what they see and hear, and retain 70 percent of what they see, hear and do (Whitehouse, 1995). Adding embedded simulation is another multimedia element that adds to the richness of the CBT experience

by involving the student at a deeper level in the learning process thus promoting retention. Simulations may be an important adjunct to learning tasks that depend upon more hands on involvement with portions of the operational environment.

The fundamental function of the simulator is to store, process, and display information about a real system and it's operating environment, to which the operator would be exposed if he were operating or practicing in the real world (Stark, 1989). High texture density is a prerequisite for perceiving optical flow discontinuities that define hills and ridges during low–level flight. Terrain contour is an especially salient feature of real–world scenes and should probably be included in simulator scenes. (Kleiss, 1992) The fidelity of the simulated environment is dependent upon the type of learning experience and skill set that the CBT is designed to enhance. The CBT lacks the computing power and fidelity of a simulator, but some elements of simulator technology can be supported within a CBT.

Two central issues dominate the choice between using a CBT, simulator or using actual equipment within the operating environment: the effectiveness and cost of using the CBT or simulator for training versus using the actual equipment for the same purposes. Effectiveness is the level of performance achieved by using artificial means to acquire the skills and training to operate the actual equipment. Cost means all cost, to include life cycle, procurement and operating costs.

The aviation night vision operational environment is a dangerous one. The ability to train for this environment is highly desirable, more so if it can be done effectively at a reasonable cost and with nominal risk. This makes simulators and CBT's attractive. The authors mission was to see if an effective low–cost CBT could be developed that would

support embedded interactive simulation of the night vision environment. Some of the more detailed scene elements were not be able to be presented, given the design decision to develop the system using personal computer COTS products.

III. METHODOLOGY

A. OBJECTIVE

The Visual Effects Simulation Demonstration (VESD) discussed here is a stand–alone "proof–of–concept" demonstration that operates on an IBM compatible PC running the Windows 95, or above, operating system. The objective of this demonstration was to develop a low fidelity simulation that would ultimately be integrated with a previously computer–based ANVIS/HUD trainer.

B. TRAINING / FUNCTIONAL REQUIREMENTS AND SPECIFICATIONS

1. Training Requirement

Earlier development work on the ANVIS/HUD trainer such as that completed by Meza (March, 1995), Kern and Shaffer (March, 1996), and Foster and Price (December, 1996) provide a foundation for specifying requirements for the Visual Effects Simulation Demonstration (VESD). These earlier studies helped establish training requirements specification and to build a baseline ANVIS/HUD trainer. The earlier versions of ANVIS/HUD (discussed by Foster and Price, 1996) incorporated text, graphics, and selected non–interactive video examples. A training feature that incorporated ability for the aviator under instruction to view HUD symbology under dynamic simulated flight conditions was desired. The capability for the trainee to visualize flight over terrain, in even the most rudimentary way, was considered to have much potential benefit to the realism and value of the ANVIS/HUD trainer. The VESD was developed to provide ANVIS/HUD trainees the ability to visualize flight over terrain while viewing dynamic changes in HUD symbology. For example, the trainee could use the VESD to observe

the effects of display clutter, by toggling selected symbology ON/OFF during the flight over terrain interactive movie demonstration.

2. Functional Specifications

The functional specifications for the demonstration were based upon a report by Ciavarelli, Baer and Sengupta (December 1994). The following table, extracted from this report, served as our initial functional specification for constructing the demonstration:

Table 3.1 Desired Functional Capabilities of a Part–task Visual Simulator

1. DISPLAY DYNAMIC OUT OF COCKPIT SCENES
    - Provide selectable terrain variations
    - Provide selectable illumination levels
    - Generate artificial light effects
    - Generate atmospheric attenuation effects
    - Generate aircraft lighting effects
2. DISPLAY BASIC FLIGHT INSTRUMENTS
    - Display primary flight instruments
    - Display selected engine/other instruments
    - Generate auditory and visual warnings
3. PROVIDE BASIC FLIGHT CONTROLS
    - Provide manual control inputs (Joystick, Throttle)
    - Provide control and display inputs
    - Provide aerodynamic response
4. PROVIDE INSTRUCTOR OPERATOR CONTROL
    - Provide scenario construction and storage

- Provide pre–programmed flight profiles
- Provide Instructor Controls and Displays

Functional capabilities for the VESD were selected from Table 3.1 and implemented using the PC hardware and software systems described above, and following procedures as outlined next.

C. HARDWARE AND SOFTWARE DESIGN

1. Hardware

The hardware consisted of an IBM Compatible PC, manufactured by Micron Inc., and the VFXI™ Helmet–mounted Display (HMD) manufactured by Forte Technologies. The PC was a 266 Mhz Pentium Pro II, configured with 128 megabytes of RAM and a 6.4 gigabyte hard disk drive. A Hercules Dynamite™ 128/4–video card was selected as the most compatible video card to be used with the HMD.

The HMD came equipped with its own ISA compatible circuit card. A VESA bus connector is used to link the VFXI card directly to the VESA feature connector on the video card. The helmet includes two LED viewing devices and headphone speakers linked to the computer via the VFX1 card. The HMD is capable of displaying a screen resolution up to 640 by 480 pixels at 60Hz refresh rate. Detailed specifications for the VFXI HMD are presented in Appendix A.

2. Software

Vista pro 4.0™ , manufactured by Romtech was used to generate Night Vision Goggle (NVG) scenes. This software program has the capability of importing Digital Elevation Models (.dem files) produced by the U.S. Geological Survey. Vista Pro can be used to create images populated with user selected terrain cover and user defined color

schemes. The software also allows the user to specify a route through the terrain, to select a type of vehicle to be used, and to specify the number of frames per second needed to create a series of bitmap images.

Following generation of sample visual scenes, the demonstration itself was created using Macromedia's Director 6.0 ™ multimedia authoring suite. This software has the capability, among other authoring features, to create interactive movies. DirectControl™ from DirectXtras was used to provide joystick capability to Director 6.0 ™. The entire Visual Effects simulation for the ANVIS/HUD CBT was developed as a set of interactive movies using the movie creation feature.

D. SYSTEM DEVELOPMENT PROCEDURES

The development capabilities of Vista Pro and Director 6.0 ™ are clearly specified in the software documentation, and adequately cover topics related to the construction of images, importing images, creating movie sequences, and the use of text and graphics editors used in the construction of a demonstrations such as that created for this thesis project. The following outlines the steps used in the construction of the VESD:

1. Creation Of Background Imagery

Using Vistapro, the following steps were used to construct the movie background imagery for the demonstration:

- Open the terrain to be viewed (. dem file).
- Select the color option and turn all colors to green to approximate ANIVIS–9 imagery for all terrain textures and tree types.
- Select lighting and environmental conditions as desired. Define flight path to be flown by helicopter (by specifying selectable waypoints). On the Path

menu the Speed needs to be calibrated to the desired airspeed. Using a conversion factor of 50.8 cm/(seconds * Knots) converts speed in knots and frame rate in frames per second (fps) to centimeters between frames, which are the required units for this field. An airspeed of 150 Knots at a frame rate of 15 fps converts to:

3.1 150 Knots * 50.8 cm/(seconds * Knots) / 15 fps = 508 cm/frame

- Select image size, which in this case is very critical. The HMD limiting dimension is 480 pixel height that must correspond to a 40 degree field–of–view. Therefor width was also constrained to 480 pixels. In order to calibrate the image to achieve angular correlation between head movement and image movement the following computation was necessary:

  3.2 480 pixels / 40 degrees = 12 pixels/degree

  The image size was chosen to be 90 degrees wide by 60 degrees high, thereby yielding

  3.3 Height = 90 degrees * 12 pixels/degree = 180 pixels

  3.4 Width = 60 degrees * 12 pixels/degree = 720 pixels

- Animate a bitmap (. bmp) series of images at 15 frames per second.
- Save and print the script file from the run.

2. Movie Generation

Using Director 6.0 ™, import imagery into a movie cast as a 32–bit color image as follows:

- Create a new eight bit custom color green scale palette in Director for use by converting images from 24–bit color Vistapro bitmaps to eight bit color

image. This step was done in the Director's Paint program. A custom palette was defined by setting the central color (Number 126) to the RGB values corresponding to the wavelength emitted by the p–40 phosphor, 560 nanometers, used in the ANVIS–9 image intensifying tubes (Red = 180, Green = 255, Blue = 0). The color zero was defined as white and color 255 was defined as black. Using the blend function, tints and shades of the central color (from white to black) were created to fill the rest of the palette.

- Transform each bitmap image from 32–bit color to eight bit color. This transformation achieves a near lossless compression of approximately three to one from the original image, and results in a closer correlation of the imagery colors to the ANVIS–9 environment.
- Create each HUD symbol component using a graphic editor, import each symbol into the cast and transform them to the custom palette.
- Select a black stage with dimensions 480 by 480.
- Import the entire bitmap series of background images into the score of the movie using the cast to time feature.
- Create a black mask with a white center circle to place over the movie on the stage to give the round appearance of the ANVIS–9 tubes. Lingo commands to respond to HMD movement were attached to this particular cast member.
- The HUD symbology was then animated to correspond to the actions in each frame, using the Script file from the Vistapro run.
- Lingo script was added to control the movie and interaction. (See appendix B).

- A menu–driven user interface was developed to allow the user to set preferences and to operate the movie sequence.
- A stand alone executable "projector" was created.
- Reset the system display to 60Hz, 256 color, 640 X 480 mode.
- View movie through the headset (or use SVGA monitor).

IV. RESULTS AND DISCUSSION

A. OVERVIEW OF VESD TRAINER

The demonstration was designed to run on an IBM compatible computer running Windows 95 or above. The hardware configuration suggested for use of the demonstration is a Pentium Pro II 266 MHz processor with 128 Mbytes RAM and at least 1.5 Gigabytes of available hard disk space. Use of the Hercules Dynamite™ 128 video card is recommended in order to meet HMD compatibility requirements. The demonstration can be viewed on an SVGA monitor, as well as the recommended HMD (Forte VFX1).

The demonstration was constructed to include several different modes of operation, and with different terrain and light scene combinations. Figure 4.1, shows the user selections available on the demonstration. The demonstration includes, (1) HUD Symbology Tutor (2) Flight without visual scan (3) Hover with Visual Scan, and (4) Flight with visual scan. As shown in Figure 4.1, the user may select various visual scene options. For example, under Flight without visual scan the user may select Desert Bright, Desert Medium or Desert Shadowed. A mountain ridge scene option is provided under the Hover with visual scan mode. Figures 4.2 and 4.3 show snapshots of Desert Bright, and Mountain Ridge Bright scenes.

The main thrust of this thesis project was to develop out–of –cockpit scenes, and to superimpose ANVIS/HUD symbology. However, during the course of our development we added the HUD symbology Tutor that can be used to review the location

Figure 4.1. Main Menu of Flight Options

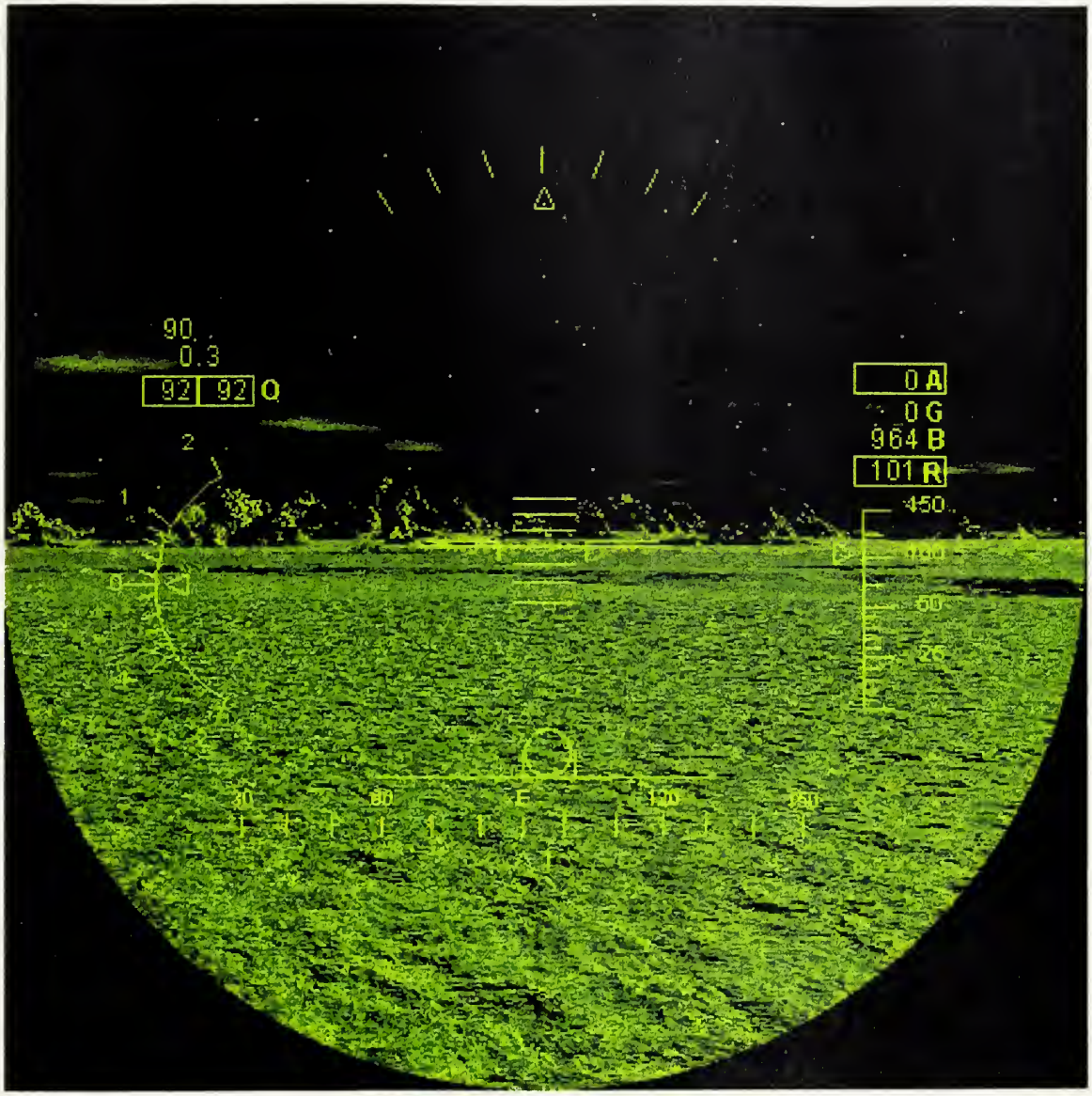

Figure 4.2. Desert Bright Scene

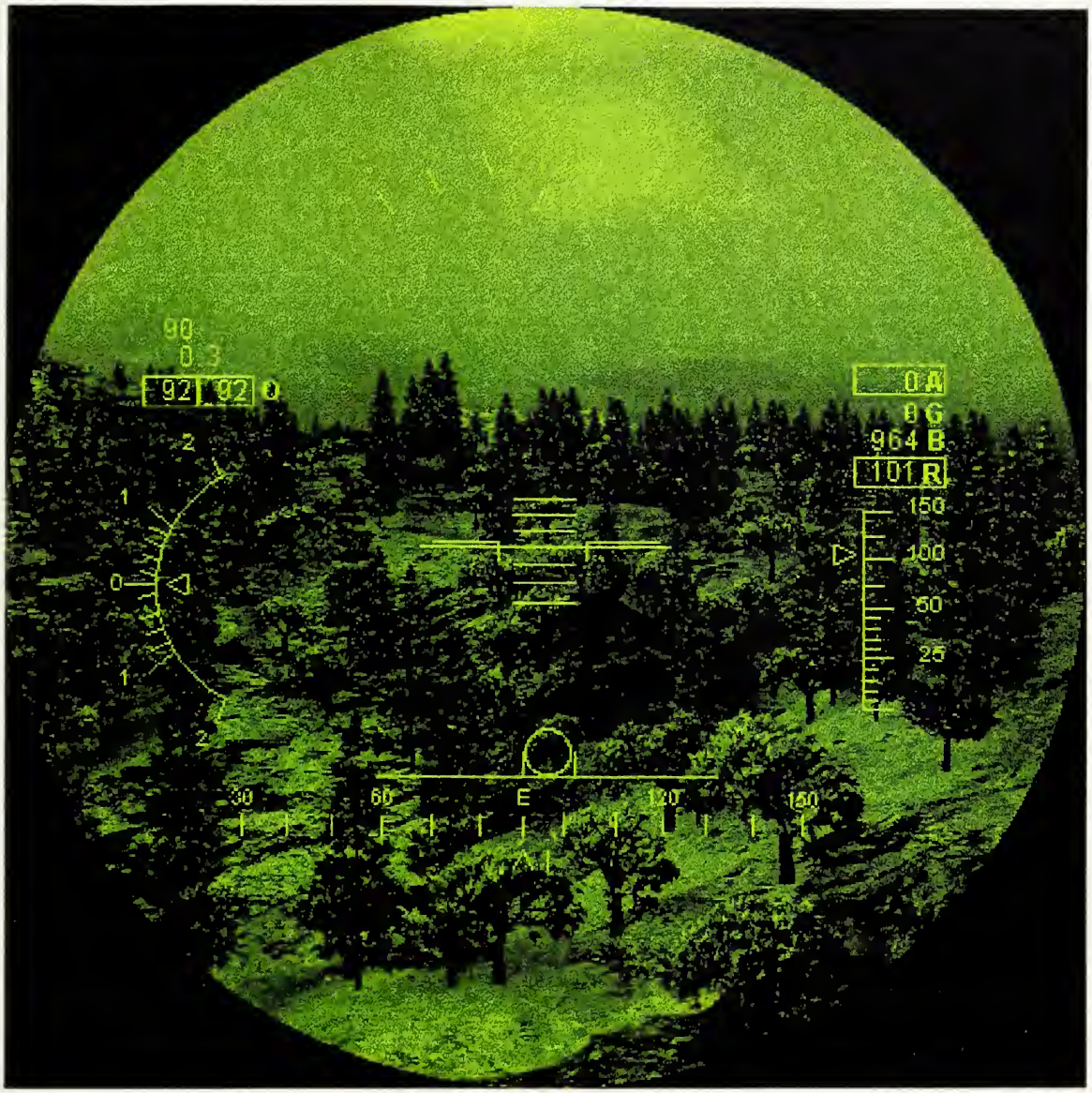

Figure 4.3. Mountain Ridge Bright Scene

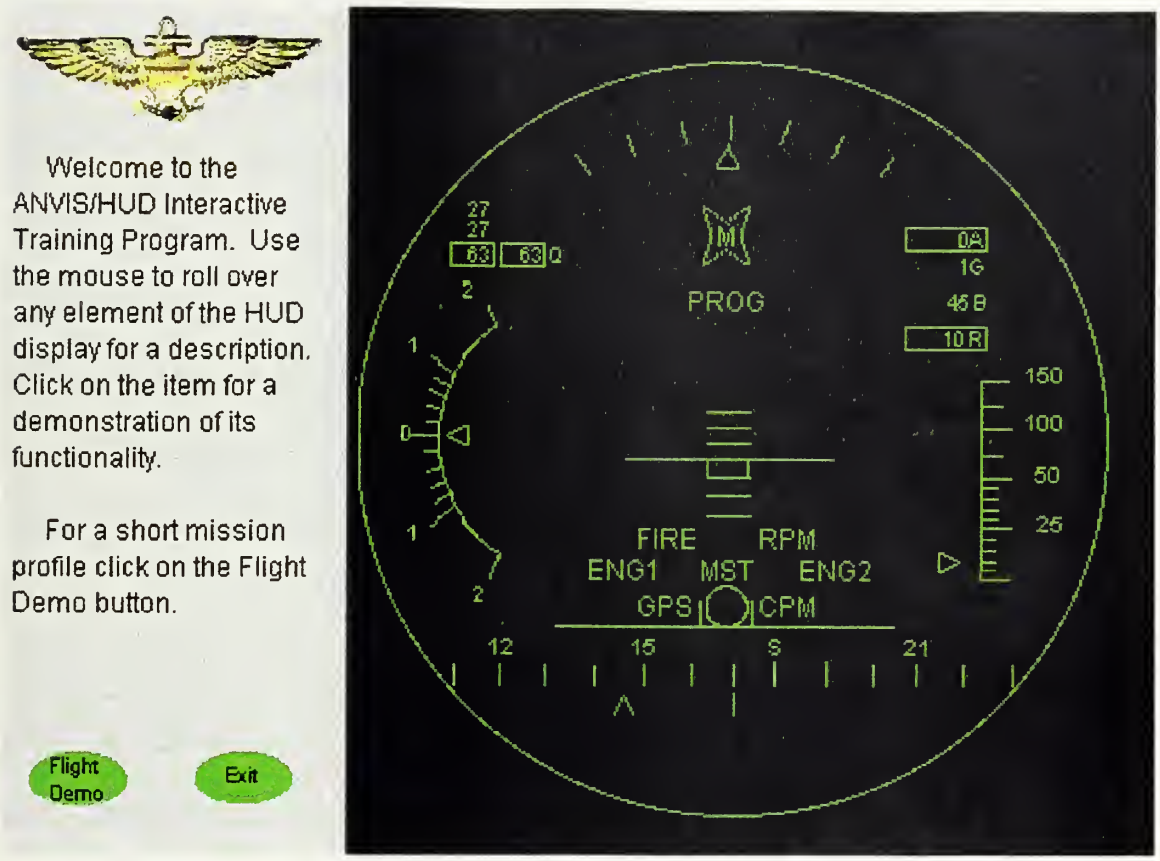

Figure 4.4. HUD Tutor Symbology Display Screen

and function of ANVIS/HUD symbols. The HUD Tutor was developed as a Shockwave™ application for use over the World Wide Web. The Tutor demonstration shows an image of the entire HUD symbology set. The user can obtain a description of any HUD symbol by moving the mouse cursor over a particular symbol. Figure 4.4 shows the HUD Tutor Symbology display screen.

B. OPERATION OF THE DEMO

The setup and software configuration instructions are presented in Appendix C. From the User's point of view, the Visual Effects Simulation Demonstration begins with the CONTENTS screen, presented in Figure 4.1. The user selects an option from the menu tabs presented in this figure. Tabs for three pages are presented: **Introduction,**

Contents, and **Exit**. The Introduction is a brief overview of the program. The Contents is a list of movie clips available with a set of hot links to executable files. Each movie selection is indicated by a button and text name. Rolling over the button causes a screen shot thumbnail of the movie scene to pop up and rolling over the text causes a short text description of the movie to pop up. The user selects a particular movie by clicking the mouse on either the text or movie thumbnail image. Each movie clip has a text description associated with it, and represents a particular lesson module. After the user is done reading the movie's text, the user can select the HUD symbology screen by left clicking on the mouse. The HUD Symbology Selection Screen is presented in Figure 4.5. The user can toggle various selected symbology items on or off. Symbols turned off will not display during the movie sequence. The HUD symbology selection screen can be activated any time during the movie presentation. The user starts the movie by clicking on the screen the **Start** or **Resume** screen controls shown in Figure 4.5. Instructions for later interaction with the movie are displayed while the movie file is loading, as shown in Figure 4.6.

If the user intends to view the movie with the HMD, then this is a good point in the process to lower the HMD goggles and keep a hand on the mouse. During the movie the mouse should not be moved. Moving the mouse while the movie is playing will cause the mouse cursor to move from the movie stage. Once off this stage, the mouse will not be active as control input while the movie is running. With the mouse in a fixed position, and cursor on stage, the user can pause the movie by clicking down and holding the right mouse key. The user may also return to the symbology selection screen by a left mouse

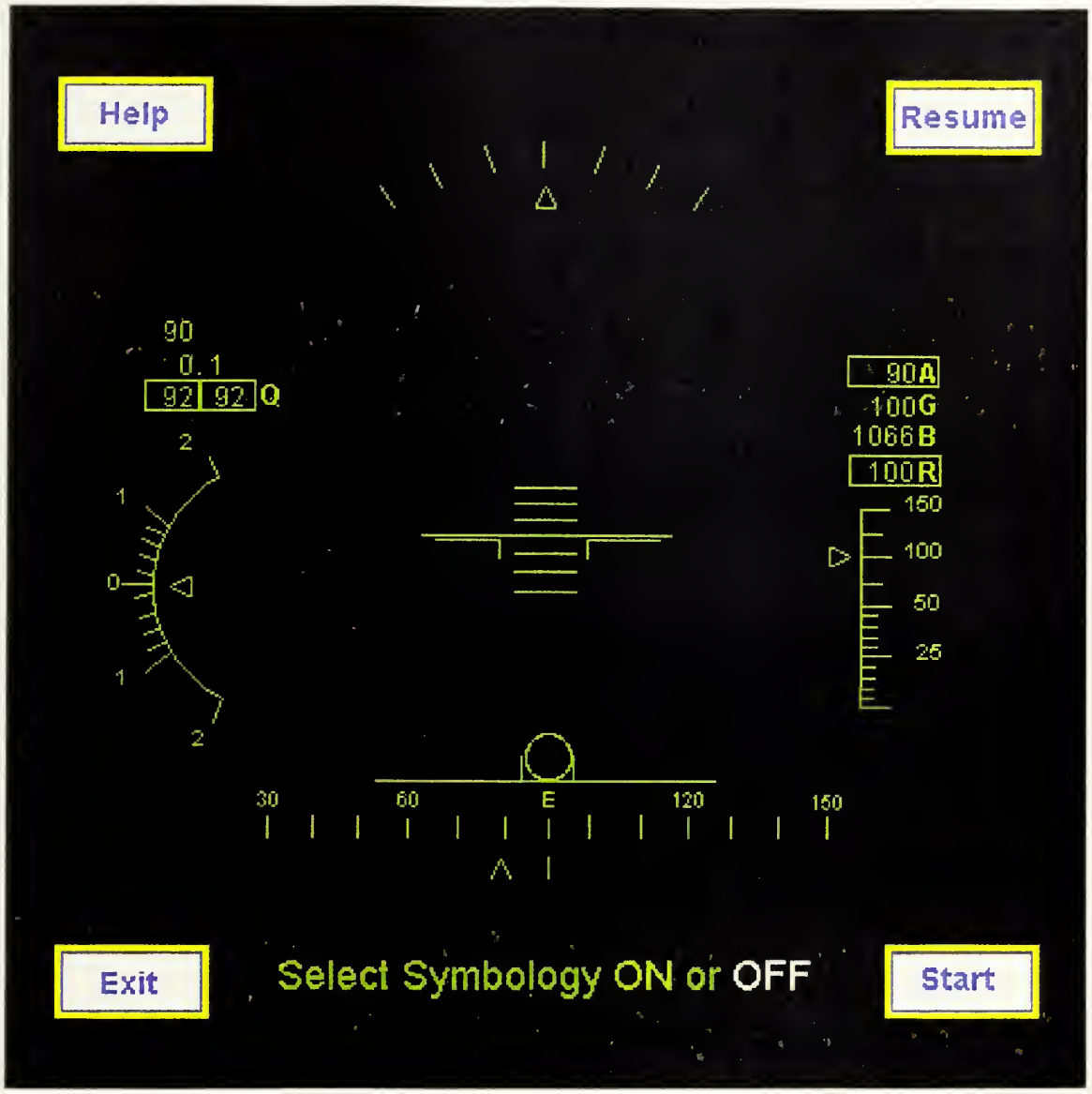

Figure 4.5. HUD Symbology Menu

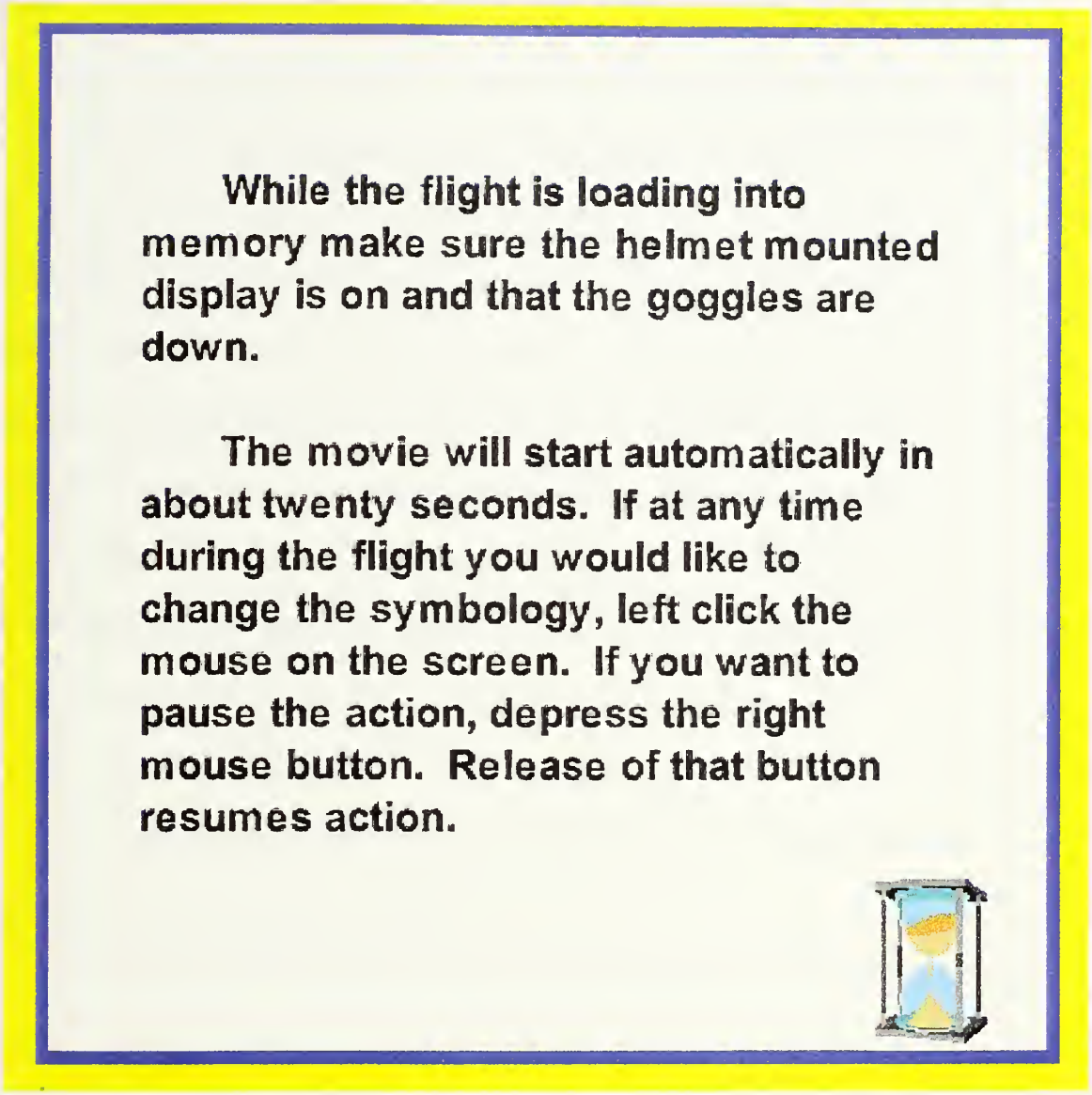

Figure 4.6. Instructions for Interactive Flight

click. Once back to the symbology screen, the user can re–configure the movie for another run. The user may also exit the movie from this particular screen.

As mentioned in the overview, there are various modes of operation in the VESD. If the movie is running in an interactive mode the user will be able to scan the image within the field of view of the movie presentation window. As in ANVIS, a 40 degree field of view is presented. The total available scene is 90 degrees horizontal and 60 degrees vertical. If the HMD is properly calibrated and the user begins the movie looking straight ahead, then the image will be centered on the screen. Head movements up, down, left or right will produce corresponding changes in the image being viewed. Visual scan is available in both Flight and Hover movie selections, but this feature reduces the flight length to about eight seconds.

C. LESSONS LEARNED

During the course of producing the VESD a number of limitations were encountered and tradeoffs were made.

The Director 6.0™ authoring environment was initially developed to produce QuickTime™ videos for the Apple environment. A vestige of that frame of reference is that Director only plays back images in bitmap (.bmp) format. Bitmaps require the most storage space of any available image format. If Director could take advantage of some form of compression it would greatly decrease disk access time. This is significant because it appears that Disk access time is the primary limiting factor in playback speed. In order to achieve continuous (no jitter) playback the entire movie must be able to be loaded into RAM or the images must be small enough and the playback rate slow enough to allow run time retrieval. Director also has no ability to conduct more than one process

simultaneously. It would be desirable if Director could look–ahead and fetch the next frame not yet in memory as soon as it has executed a frame that will not be displayed again while it is displaying other images. Currently refresh rates of 28 frames per second are achievable for the interactive flight in this demonstration using the recommended system configuration. Due to the requirement to have all of the frames loaded into memory only about eight seconds of 15 fps video can be loaded into the available RAM. Because the hover flights only display a single frame, higher refresh rates (60 Hz) are achievable.

It was found that the generation of bitmap sequences for the background imagery using Vistapro could not be done on multiple computers for the same route. Consistent image registration for sequences of imagery generated on other computers is not guaranteed. The entire sequence must be generated on a single machine.

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Given the current state of technology it is possible to develop small–scale simulation on a Personal Computer. Using low cost software we were able to produce reasonably good quality NVG and HUD imagery that could be animated to produce a somewhat lifelike simulation.

The primary limitation in developing and running the application proved to be storage and access. Because Director can not generate a request for imagery while it is performing other functions the program must wait for imagery when it needs it. This directly impacts playback performance as soon as the imagery stored in RAM is exhausted. If imagery could be stored in a compressed format it would require less time to retrieve. This would be helpful as long as expansion was reasonably fast. Again if Director could take advantage of the parallel processing capability under Windows 95 this could be done while imagery is being displayed or manipulated. Given the large size of bitmap format files even disk storage becomes a factor. The fully interactive module with only about 15 seconds of imagery requires nearly 100 Megabytes of memory. At that size only about six or seven modules could be stored on a single CD ROM uncompressed.

The VESD definitely has the capability of giving the user a good idea of what HUD symbology looks like when superimposed over terrain in an NVG environment. The ability to experiment in a safe and controlled environment with different levels of HUD symbology decluttering is clearly offered. It also has the capability of being run on

systems that would be available in the squadron environment. The VESD could prove to be a valuable addition to current ANVIS/HUD CBT systems.

Incorporation of the VESD into the ANVIS/HUD trainer would allow fleet units to experiment with the HUD symbology in a safe, low cost, high availability environment. Given additional advances in technology we may no longer need a dark room, good weather, nighttime, expensive equipment or an aircraft to play with different configurations of the HUD symbology. The total cost of this system was a fraction of the cost of a single ANVIS–9 unit.

B. RECOMMENDATIONS

During the course of development it was not possible to fully explore all of the avenues available as fully as we would have liked. Some of the more interesting areas that we could not explore are discussed here.

- Use a higher speed personal computer. 300 MHz Pentium Pro II computers running on 100 MHz motherboards should be available soon. Combining a faster computer with a faster storage retrieval system would increase performance. The use of RAID (Redundant Array of Inexpensive Disks) technology may improve retrieval rates. If the per frame retrieval rate could be increased sufficiently the only limitation to the length of the flight would be moved from the amount of RAM to the size of the hard disk.
- Add voice explanations to the audio track to point out or explain scene elements. Voice instruction can also describe various HUD symbology display mixes given the prevailing lighting conditions and environmental type.

- Voice recognition Xtras (modules) are available for Director 6.0 ™. Incorporating the capability to provide hands free operation might be useful. This would be significantly easier while wearing the HMD. Since voice recognition is computationally intensive it would most likely require sacrifices in image processing.
- Develop the system from scratch using a package that allows the programmer to have greater control over memory management and to optimize code for greater processing efficiency.
- There are second source plug–ins that allow Director 6.0 ™ to access three–dimensional imagery. The possibility of having a 360 degree viewing area would increase the realism of the simulation
- This application could be developed for use over the World Wide Web. Director 6.0 ™ applications can be converted to Macromedia's own Shockwave format that works very well on the WWW. The application would require significant bandwidth to run but we believe that the internet bandwidth limation will ultimately be overcome.
- Use a better HMD to get better resolution. The trend in hardware costs is to come down. A higher quality resolution HMD may be available in the near future at costs similar to the present cost of the VFX1.

Continued development of demonstrations like this one that depict night vision scenery many ultimately lead to transportable desk–top trainers. Such devices, given advances in technology, may be a valuable adjunct to the Navy's night vision training program. For example, if visual simulator fidelity improves in such desk–top trainers,

some of the NITE lab demonstrations may be possible to do "electronically". Other future applications include use of such simulation in the night vision introductory and refresher courses.

LIST OF REFERENCES

Ciavarelli, Anthony, Kishore Sengupta, and Wolfgang Baer, *Night Vison Goggle (NVG) Training Technology Report*, Technology Report, Naval Postgraduate School, Monterey, CA, December 1994.

ED340388, ERIC Digest, Office of Educational Research and Improvement, 1991.

Epperson, Sean T., *Animation within a Multimedia Training System for Night Vision Goggles*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1995.

Foster, Robert W., and Alfred B. Price, Jr., *Instructional Design of Computer–Based Training*, Master's Thesis, Naval Postgraduate School, Monterey, CA, December 1996.

Kern, Dabney R., and Kenan J. Shaffer, *Re–Engineering a Computer–Based Trainer for a Helicopter Night Vision System*, Master's Thesis, Naval Postgraduate School, Monterey, CA, September 1996.

Kirkland, Sonny, *Introduction to Multimedia: An Overview for Educators*, Available at: http://cee.indiana.edu/workshops/multipres/MM.html, July 1997.

Kleissd, J. A., *Tradeoffs Among Types of Scene Detail for Simulating Low–Altitude Flight*, Conference Paper, 1992 IEE International Conference on Systems, Man and Cybernetics, IEEE, New York, NY, 1992.

Meza, Francisco Q., *Developing Multimedia Instructional Systems: An Example Application for Training in Night Vision Goggles*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1995.

Nixon, Daniel E., *Development and Application of a Multimedia Assessment Tool*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1997.

Obinger, Diana, *Introduction to Multimedia Instruction: A Primer*, University of North Carolina Press, 1992.

Orlanski, Jesse, Carl J. Dahlman, Colin P. Hammon, John Metzko, Henry L. Taylor, and Christine Youngblut, *The Value of Simulation for Training*, IDA Paper P–2982, Institute for Defense Analyses, Alexandria, VA, September 1994.

Stark, Edward A., *Simulation*, In: Richard S. Jensen, *Aviation Psychology*, Gower Technical, Vermont, VA, 1989.

Whitehouse, D., and G. Pellegrin, *Computer Based Training, Is it Worth the Money?*, Conference Paper, Conference Record of 1995 Annual Pulp and Paper Industry Technical Conference, IEEE, New York, NY, 1995.

Xie, Geoffrey, *Multimedia*, Lecture Notes (unpublished). Monterey, CA, 1997.

APPENDIX A. FORTE VFX1™ SPECIFICATIONS AND REQUIREMENTS

3D Stereoscopic Flip-Up Visor

- With focus and IPD adjustments
- High contrast color video
- Dual 0.7" color liquid crystal displays
- 789(RGB) x 230 resolution x 2 263 x 230 True Pixels
- 181,470 pixels
- 256 Colors
- Works with existing, standard VGA applications
- Optics
- Field of view:
- 26.4 degrees vertical
- 35.5 degrees horizontal
- 45.0 degrees diagonal
- High quality color corrected optics
- 45 degree field of view
- Provides large screen stereoscopic imaging
- Virtual Orientation System (VOS) Sourceless Head Tracker
- 60Hz Refresh Rate (60 times per second)
- With three degrees of freedom:
    - Yaw (azimuth): 360 degrees
    - Pitch (elevation): +/- 70 degrees

- Roll (tilt): +/- 70 degrees
- VFX1 HEADGEAR Interface Protocol (VIP) Card
- Interfaces with standard VGA card and sound card
- Includes ACCESS.bus host
- Connect up to 125 different devices
- Does not require serial port connection

CyberPuck Controller

- Left or right-handed control
- ACCESS.bus device
- 3 programmable buttons

High Quality Stereo Headphones

- Designed by AKG of Austria
- 20 Hz - 20 kHz
- Stereo and 3D sound capable

Comset Microphone

- Hands free communication
- Gamer to gamer
- Voice recognition capable

Ergonomic Design

- Comfort
- Control
- Extended use

Power Requirements

- 5 volts, 2.0 watts
- 0.4 amps (with Cyberpuck controller)
- Does not require external power sources

System Requirements

- IBM PC or compatible, 386, 486 or Pentium
- Video card (VGA minimum) with 100% VESA Standard Compliant Feature Connector
    - ISA
    - VLB
    - PCI
- Free ISA Expansion Slot
- Does Not use a COM/Serial Port
- One Free Base Port (choices):
    - 260
    - 280
    - 2A0
    - 2C0
- One Free IRQ (choices):
    - 5
    - 7
    - 10
    - 11
    - 12
    - 15

- Stereo sound card (for audio)

APPENDIX B. LINGO SCRIPT

The following is a complete listing of the Lingo Code for the modules Hudmain and Interactive. Hudmain is the menu from which modules are chosen. Interactive is the flight with full HMD interactivity. The code for the other modules is very similar to the code in one of these two modules, primarily Interactive.

A. Lingo Script for Interactive (Desert Bright with Visual Scan)

Script 2 Begin

```
on exitFrame
    go to "Beginning"
end
```

Script 3 Go To HUD Menu

```
on mouseUp
    go "Hud Menu"
end
```

Script 6 Set Stage & Preload Flight

```
on enterFrame

    set the visible of sprite 24 to gAttitude
    set the visible of sprite 25 to gAttitude

    set the visible of sprite 8 to gRoll
    set the visible of sprite 9 to gRoll

    set the visible of sprite 27 to gBarAlt
    set the visible of sprite 28 to gBarAlt
    set the visible of sprite 29 to gBarAlt
    set the visible of sprite 30 to gBarAlt

    set the visible of sprite 11 to gBall
    set the visible of sprite 12 to gBall

    set the visible of sprite 44 to gDigWayPt

    set the visible of sprite 14 to gDigRadAlt
    set the visible of sprite 15 to gDigRadAlt
```

```
        set the visible of sprite 32 to gAirspeed
        set the visible of sprite 33 to gAirspeed

        set the visible of sprite 35 to gGroundSpd
        set the visible of sprite 36 to gGroundSpd

        set the visible of sprite 38 to gTorque
        set the visible of sprite 39 to gTorque
        set the visible of sprite 40 to gTorque
        set the visible of sprite 41 to gTorque
        set the visible of sprite 42 to gTorque

        set the visible of sprite 46 to gTimeToGo
        set the visible of sprite 47 to gTimeToGo
        set the visible of sprite 48 to gTimeToGo

        updatestage
        unload
        preload 10, 260
    end

    on exitframe
        go to frame 10
    end exitframe
```

Script 15 RadAlt Toggle

```
    Global gRadAlt

    on mouseDown

        puppetSound "click"

        if gRadAlt = True then

            set gRadAlt = False
            set the memberNum of Sprite 5 to 699
            set the memberNum of Sprite 6 to 692
            puppetSprite 5, True
            puppetSprite 6, True

         else
```

```
            set gRadAlt = True

            puppetSprite 5, False
            puppetSprite 6, False

        end if

        updatestage

        startTimer
        repeat while the timer < 15
            nothing
        end repeat

        puppetSound 0

    end mouseDown
```

Script 32 Circle

```
    Global gCurrentFrame, gPause, HMD_Center_X, HMD_Center_Y
    Global Window_Size, Pixel_Factor

    on mouseUp

        set gCurrentFrame = the Frame

        --remember the frame from which the
        --user leaves the movie to return to the HUD symbology menu
        --By clicking anywhere on the screen the user
        --will be returned to the symbology selection menu

        go to "HUD Menu"
    end

--The following script implements a PAUSE  to the movie that
--retains interactivity.  It is executed on RIGHT MOUSE DOWN

    on RightMouseDown

        set gPause to True   --Request to Pause is set true

    end OnRightMouseDown
```

```
on exitFrame

    set HMD_X = joyGetPos (1,1)  -- get the X coord of headset
    set HMD_Y = joyGetPos (1,2)  -- get the Y coord of headset

    set the locH of sprite 1 to ((HMD_X - HMD_Center_X)/Pixel_factor) +
        Window_Size
    set the locV of sprite 1 to ((HMD_Y - HMD_Center_Y)/Pixel_factor) +
        Window_Size

    if gPause Then

        go to the frame    --While pausing keep displaying the same frame

    end If

end onexitframe

on RightMouseUp

    set gPause to False   --Turn off request for Pause

end OnRightMouseUp
```

Script 34 Initialize Variables

```
Global gCompass, gRadAlt, gVSI, gAttitude, gRoll, gBarAlt, gBall, gWayPt
Global gDigRadAlt, gDigWayPt, gAirspeed, gGroundSpd, gTimeToGo
Global gTorque, gCurrentFrame
Global Window_Size, Pixel_Factor, HMD_Center_X, HMD_Center_Y
Global gPause  --vairable to pause movie during execution

on EnterFrame

    set gPause    to  False  --Movie is NOT PAUSED at the beginning
    set gCompass  to  True  -- Set Compass to be displayed
    set gRadAlt   to  True
    set gVSI      to  True
    set gAttitude to  True
    set gRoll     to  True
    set gBarAlt   to  True
    set gBall     to  True
    set gWayPt    to  True

    set gCurrentFrame to 4
```

```
set HMD_Center_X = 32767
set HMD_Center_Y = 32767
set Window_Size = 240
set Pixel_Factor = 15.1

set gDigWayPt to  True
set gDigRadAlt to True
set gAirSpeed to  True
set gGroundSpd to True
set gTorque   to  True
set gTimeToGo to  True

set the visible of sprite 17 to gCompass
set the visible of sprite 18 to gCompass
set the visible of sprite 19 to gWayPt

set the visible of sprite 5 to gRadAlt
set the visible of sprite 6 to gRadAlt

set the visible of sprite 21 to gVSI
set the visible of sprite 22 to gVSI

set the visible of sprite 24 to gAttitude
set the visible of sprite 25 to gAttitude

set the visible of sprite 8 to gRoll
set the visible of sprite 9 to gRoll

set the visible of sprite 27 to gBarAlt
set the visible of sprite 28 to gBarAlt
set the visible of sprite 29 to gBarAlt
set the visible of sprite 30 to gBarAlt

set the visible of sprite 11 to gBall
set the visible of sprite 12 to gBall

set the visible of sprite 44 to gDigWayPt

set the visible of sprite 14 to gDigRadAlt
set the visible of sprite 15 to gDigRadAlt

set the visible of sprite 32 to gAirspeed
set the visible of sprite 33 to gAirspeed
```

```
        set the visible of sprite 35 to gGroundSpd
        set the visible of sprite 36 to gGroundSpd

        set the visible of sprite 38 to gTorque
        set the visible of sprite 39 to gTorque
        set the visible of sprite 40 to gTorque
        set the visible of sprite 41 to gTorque
        set the visible of sprite 42 to gTorque

        set the visible of sprite 46 to gTimeToGo
        set the visible of sprite 47 to gTimeToGo
        set the visible of sprite 48 to gTimeToGo

    end

    on exitframe
        go to the frame
    end exitframe
```

Script 60, Resume Button Script

```
    on mouseUp

        Global gCompass, gRadAlt, gVSI, gAttitude, gRoll, gBarAlt, gBall
        Global gDigRadAlt, gDigWayPt, gAirspeed, gGroundSpd, gTimeToGo
        Global gTorque, gWayPt,  gCurrentFrame

        -- Set each of the sprites to display or turn off the HUD symbol
        -- it contains based on user selections from the HUD Menu

        set the visible of sprite 17 to gCompass
        set the visible of sprite 18 to gCompass
        --Can't have the Waypointer without the compass
        set the visible of sprite 19 to (gWayPt AND gCompass)

        set the visible of sprite 5 to gRadAlt
        set the visible of sprite 6 to gRadAlt

        set the visible of sprite 21 to gVSI
        set the visible of sprite 22 to gVSI

        set the visible of sprite 24 to gAttitude
        set the visible of sprite 25 to gAttitude

        set the visible of sprite 8 to gRoll
```

```
        set the visible of sprite 9 to gRoll

        set the visible of sprite 27 to gBarAlt
        set the visible of sprite 28 to gBarAlt
        set the visible of sprite 29 to gBarAlt
        set the visible of sprite 30 to gBarAlt
        set the visible of sprite 11 to gBall
        set the visible of sprite 12 to gBall

        set the visible of sprite 44 to gDigWayPt

        set the visible of sprite 14 to gDigRadAlt
        set the visible of sprite 15 to gDigRadAlt

        set the visible of sprite 32 to gAirspeed
        set the visible of sprite 33 to gAirspeed

        set the visible of sprite 35 to gGroundSpd
        set the visible of sprite 36 to gGroundSpd

        set the visible of sprite 38 to gTorque
        set the visible of sprite 39 to gTorque
        set the visible of sprite 40 to gTorque
        set the visible of sprite 41 to gTorque
        set the visible of sprite 42 to gTorque

        set the visible of sprite 46 to gTimeToGo
        set the visible of sprite 47 to gTimeToGo
        set the visible of sprite 48 to gTimeToGo

        --Turn off the button and return to the place in the movie
        --from which the HUD symbology menu was called.

        puppetsprite 51, False
        go to gCurrentFrame


    end onMouseUp

Script 61  Start

     on mouseUp

        puppetsprite 49, False
        go to frame 4
```

```
end
```

Script 62 Exit text

```
on mouseUp

    puppetsprite 50, False
    play done --end movie

end
```

Script 71 Return to HUD Menu

```
Global gCompass, gRadAlt, gVSI, gAttitude, gRoll, gBarAlt, gBall, gWayPt
Global gDigRadAlt, gDigWayPt, gAirspeed, gGroundSpd, gTimeToGo
Global gTorque
Global Rollover_sound

on enterFrame

    --Turn on all the sprites so the user can see the status of all
    --of the symbols.  Green symbols are on, White is off.

    set the visible of sprite 17 to True
    set the visible of sprite 18 to True
    set the visible of sprite 19 to True

    set the visible of sprite 5 to True
    set the visible of sprite 6 to True

    set the visible of sprite 21 to True
    set the visible of sprite 22 to True

    set the visible of sprite 24 to True
    set the visible of sprite 25 to True
    set the visible of sprite 8 to True
    set the visible of sprite 9 to  True
    set the visible of sprite 27 to True
    set the visible of sprite 28 to True
    set the visible of sprite 29 to True
```

```
    set the visible of sprite 30 to True

    set the visible of sprite 11 to True
    set the visible of sprite 12 to True

    set the visible of sprite 44 to True

    set the visible of sprite 14 to True
    set the visible of sprite 15 to True

    set the visible of sprite 32 to True
    set the visible of sprite 33 to True

    set the visible of sprite 35 to True
    set the visible of sprite 36 to True

    set the visible of sprite 38 to True
    set the visible of sprite 39 to True
    set the visible of sprite 40 to True
    set the visible of sprite 41 to True
    set the visible of sprite 42 to True

    set the visible of sprite 46 to True
    set the visible of sprite 47 to True
    set the visible of sprite 48 to True

end on EnterFrame

--Update each sprite with the proper character.
--The global flags hold the status of each cast member
--True = ON and False = OFF
--IF the symbol was turned ON then it is displayed in green
--ELSE
--If the symbol is OFF then it is displayed in White

on exitFrame

    if gRadAlt = True then
        set the memberNum of Sprite 5 to  16  --Green
        set the memberNum of Sprite 6 to  8
    else
        set the memberNum of Sprite 5 to 699  --White
        set the memberNum of Sprite 6 to 692
    end if
```

```
if gRoll = True then
    set the memberNum of Sprite 8 to  11  --Green
    set the memberNum of Sprite 9 to  92
else
    set the memberNum of Sprite 8 to 695  --White
    set the memberNum of Sprite 9 to 700
end if


if gBall = True then
    set the memberNum of Sprite 11 to  12  --Green
    set the memberNum of Sprite 12 to  24
else
    set the memberNum of Sprite 11 to 696  --White
    set the memberNum of Sprite 12 to 707
end if

if gDigRadAlt = True then
    set the memberNum of Sprite 14 to  19  --Green
    set the memberNum of Sprite 15 to  26
else
    set the memberNum of Sprite 14 to 702  --White
    set the memberNum of Sprite 15 to 726
end if

if gCompass = True then
    set the memberNum of Sprite 17 to 27
    set the memberNum of Sprite 18 to  7
    set the memberNum of Sprite 19 to 28
else
    set the memberNum of Sprite 17 to 709
    set the memberNum of Sprite 18 to 691
    set the memberNum of Sprite 19 to 710
end if

if gWayPt = True then
    set the memberNum of Sprite 19 to 28
else
    set the memberNum of Sprite 19 to 710
end if

if gVSI = True then
    set the memberNum of Sprite 21 to  9
    set the memberNum of Sprite 22 to  18
else
```

```
    set the memberNum of Sprite 21 to 693
    set the memberNum of Sprite 22 to 701
end if

if gAttitude = True then
    set the memberNum of Sprite 24 to  10
    set the memberNum of Sprite 25 to  58
else


    set the memberNum of Sprite 24 to 694
    set the memberNum of Sprite 25 to 739
end if

if gBarAlt = True then
    set the memberNum of Sprite 27 to  22
    set the memberNum of Sprite 28 to  55
    set the memberNum of Sprite 29 to  42
    set the memberNum of Sprite 30 to  42
else
    set the memberNum of Sprite 27 to 705
    set the memberNum of Sprite 28 to 736
    set the memberNum of Sprite 29 to 722
    set the memberNum of Sprite 30 to 722
end if

if gAirSpeed = True then
    set the memberNum of Sprite 32 to  20
    set the memberNum of Sprite 33 to  50
else
    set the memberNum of Sprite 32 to 703
    set the memberNum of Sprite 33 to 731
end if

if  gGroundSpd = True then
    set the memberNum of Sprite 35 to  21
    set the memberNum of Sprite 36 to  26
else
    set the memberNum of Sprite 35 to 704
    set the memberNum of Sprite 36 to 726
end if

if  gTorque = True then
    set the memberNum of Sprite 38 to 30
    set the memberNum of Sprite 39 to 30
```

```
        set the memberNum of Sprite 40 to 23
        set the memberNum of Sprite 41 to 52
        set the memberNum of Sprite 42 to 52
    else
         set the memberNum of Sprite 38 to 711
         set the memberNum of Sprite 39 to 711
         set the memberNum of Sprite 40 to 706
        set the memberNum of Sprite 41 to 733
        set the memberNum of Sprite 42 to 733
    end if

    if  gDigWayPt = True then
        set the memberNum of Sprite 44 to  50
    else
        set the memberNum of Sprite 44 to 731
    end if

    if  gTimeToGo = True then
        set the memberNum of Sprite 46 to 36
        set the memberNum of Sprite 47 to 31
        set the memberNum of Sprite 48 to 37
    else
        set the memberNum of Sprite 46 to 716
        set the memberNum of Sprite 47 to 712
        set the memberNum of Sprite 48 to 717
    end if

-- Controls the action buttons on the hud display

case rollover() of

49: puppetSprite 49, True
    set the memberNum of Sprite 49 to 61
    if Rollover_sound = False then
      puppetSound "Rollover"
      set Rollover_sound = True
    end if

  50: puppetsprite 50, True
    set the membernum of sprite 50 to 62
    if Rollover_sound = False then
      puppetSound "Rollover"
      set Rollover_sound = True
    end if
```

```
    51: puppetsprite 51, True
      set the membernum of sprite 51 to 60
      if Rollover_sound = False then
        puppetSound "Rollover"
        set Rollover_sound = True
      end if



    52: puppetsprite 52, True
      set the membernum of sprite 52 to 63
      if Rollover_sound = False then
        puppetSound "Rollover"
        set Rollover_sound = True
      end if

      ---Resets the initial display and unmasks the cursor

      otherwise set the memberNum of Sprite 49 to 167
      puppetSprite 49, False

      set the membernum of sprite 50 to 200
      puppetsprite 50, False

      set the membernum of sprite 51 to 199
      puppetsprite 51, False

      set the membernum of sprite 52 to 201
      puppetsprite 52, False

      set Rollover_sound = False
      puppetSound 0
    end case

    updatestage
    go to the frame

end exitframe
```

Script 72 Roll Toggle

```
Global gRoll

on mouseDown

  puppetSound "click"

  if gRoll = True then
    set gRoll = False

    puppetSprite 8, True
    puppetSprite 9, True

    set the memberNum of Sprite 8 to 695
    set the memberNum of Sprite 9 to 700

  else

    set gRoll = True

    puppetSprite 8, False
    puppetSprite 9, False

  end if

  updatestage

  startTimer
  repeat while the timer < 15
    nothing
  end repeat

  puppetSound 0

end mouseDown
```

Script 73 Ball Toggle

```
Global gBall

on mouseDown

  puppetSound "click"

  if gBall = True then
```

```
    set gBall = False

  --  puppetSprite 11, True
  --  puppetSprite 12, True

    set the memberNum of Sprite 11 to 696
    set the memberNum of Sprite 12 to 707

    puppetSprite 11, True
    puppetSprite 12, True

   else

    set gBall = True

    puppetSprite 11, False
    puppetSprite 12, False

   end if

   updatestage

   startTimer
   repeat while the timer < 15
    nothing
   end repeat

   puppetSound 0

end mouseDown

--if gRadAlt = True then

  --set gRadAlt = False

  --set the memberNum of Sprite 5 to 699
  --set the memberNum of Sprite 6 to 692

  --puppetSprite 5, True
  --puppetSprite 6, True

--else

  --set gRadAlt = True
```

```
  --puppetSprite 5, False
  --puppetSprite 6, False

 --end if
```

Script 74 Digital Rad Alt Toggle

```
 Global gDigRadAlt

 on mouseDown

  puppetSound "click"

  if gDigRadAlt = True then
   set gDigRadAlt = False

   set the memberNum of Sprite 14 to 702
   set the memberNum of Sprite 15 to 726

  else

   set gDigRadAlt = True

   puppetSprite 14, False
   puppetSprite 15, False

  end if

  updatestage

  startTimer
  repeat while the timer < 15
   nothing
  end repeat

  puppetSound 0

 end mouseDown
```

Script 86 VR INSTRUCTIONS

```
 on exitframe
  go to frame 10
 end exitframe
```

Script 91 Compass Toggle

```
--SEE WAYPOINT TOGGLE, THE WAYPOINT TURNS OFF WITH THE
COMPASS!!!

Global gCompass, gWayPt

on mouseDown  --When user clicks on the Compass or Lubber Line

 puppetSound "click"  --Make a clicking noise

 if   gCompass = True then --If Compass is ON

  set gCompass = False    --Then turn Compass, Lubber line
  set gWayPt   = False    --and Waypoint OFF

  puppetSprite 17, True   --Take control of the Compass
  puppetSprite 18, True   --Lubber line
  puppetSprite 19, True   --and Waypoint Sprites

  set the memberNum of Sprite 17 to 709  --Make the Compass
  set the memberNum of Sprite 18 to 691  --Lubber line
  set the memberNum of Sprite 19 to 710  --and Waypoint WHITE

 else             --If the Compass was already turned OFF

  set gCompass = True    --Turn the Compass, Lubber Line
  set gWayPt  =  True    --and Waypoint back ON

  puppetSprite 17, False  --Return control of the Compass
  puppetSprite 18, False  --Lubber line
  puppetSprite 19, False  --and Waypoint Sprites to the movie


 end if

 updatestage    --Re-draw the screen to show status of symbol

 startTimer
 repeat while the timer < 15  --Delay to allow the click sound to play
  nothing
 end repeat
```

```
  puppetSound 0              --Turn off the click sound

 end mouseDown
```

Script 106 Waypoint Toggle

```
 --SEE COMPASS TOGGLE, THESE TWO ARE INTER-RELATED!!!

 Global gWayPt, gCompass

 on mouseDown

  puppetSound "click"

  if (gWayPt AND gCompass) = True then

   set gWayPt = False
   puppetSprite 19, True
   set the memberNum of Sprite 19 to 710

  else

   set gWayPt = gCompass
   puppetSprite 19, False  --Return control of the sprite to the movie

  end if

  updatestage

  startTimer
  repeat while the timer < 15  --DELAY
   nothing
  end repeat

  puppetSound 0

 end mouseDown
```

Script 107 VSI Toggle

```
 Global gVSI

 on mouseDown
```

```
    puppetSound "click"


    if gVSI = True then

      set gVSI = False
      puppetSprite 21, True
      puppetSprite 22, True
      set the memberNum of Sprite 21 to  693
      set the memberNum of Sprite 22 to  701

    else

      set gVSI = True
      puppetSprite 21, False
      puppetSprite 22, False

    end if

    updatestage

    startTimer
    repeat while the timer < 15
      nothing
    end repeat

    puppetSound 0

  end mouseDown
```

Script 108 Attitude Indicator Toggle

```
  Global gAttitude

  on mouseDown

    puppetSound "click"

    if gAttitude = True then
      set gAttitude = False

      puppetSprite 24, True
      puppetSprite 25, True

      set the memberNum of Sprite 24 to 694
```

```
    set the memberNum of Sprite 25 to 739

  else

    set gAttitude = True

    puppetSprite 24, False
    puppetSprite 25, False

  end if

  updatestage

  startTimer
  repeat while the timer < 15
    nothing
  end repeat

  puppetSound 0

end mouseDown
```

Script 121 BarAlt Toggle

```
Global gBarAlt

on mouseDown

  puppetSound "click"

  if gBarAlt = True then
    set gBarAlt = False

    puppetSprite 27, True
    puppetSprite 28, True
    puppetSprite 29, True
    puppetSprite 30, True

    set the memberNum of Sprite 27 to 705
    set the memberNum of Sprite 28 to 736
    set the memberNum of Sprite 29 to 722
    set the memberNum of Sprite 30 to 722

  else
```

```
    set gBarAlt = True

    puppetSprite 27, False
    puppetSprite 28, False
    puppetSprite 29, False
    puppetSprite 30, False

   end if

   updatestage

   startTimer
   repeat while the timer < 15
    nothing
   end repeat

   puppetSound 0

  end mouseDown
```

Script 122 Airspeed Toggle

```
  Global gAirSpeed

  on mouseDown

   puppetSound "click"

   if gAirSpeed = True then
    set gAirSpeed = False

    puppetSprite 32, True
    puppetSprite 33, True

    set the memberNum of Sprite 32 to 703
    set the memberNum of Sprite 33 to 731

   else

    set gAirSpeed = True

    puppetSprite 32, False
    puppetSprite 33, False

   end if
```

```
    updatestage

    startTimer
    repeat while the timer < 15
      nothing
    end repeat

    puppetSound 0

  end mouseDown
```

Script 123 Ground Speed Toggle

```
  Global gGroundSpd

  on mouseDown

    puppetSound "click"

    if gGroundSpd = True then
      set gGroundSpd = False

      puppetSprite 35, True
      puppetSprite 36, True

      set the memberNum of Sprite 35 to 704
      set the memberNum of Sprite 36 to 726

     else

      set gGroundSpd = True

      puppetSprite 35, False
      puppetSprite 36, False

    end if

    updatestage

    startTimer
    repeat while the timer < 15
      nothing
    end repeat
```

```
  puppetSound 0

end mouseDown
```

Script 124 Torque Toggle

```
Global gTorque
on mouseDown

  puppetSound "click"

   if gTorque = True then
   set gTorque = False

   puppetSprite 38, True
   puppetSprite 39, True
   puppetSprite 40, True
   puppetSprite 41, True
   puppetSprite 42, True

   set the memberNum of Sprite 38 to 711
   set the memberNum of Sprite 39 to 711
   set the memberNum of Sprite 40 to 706
   set the memberNum of Sprite 41 to 733
   set the memberNum of Sprite 42 to 733

  else

   set gTorque = True

   puppetSprite 38, False
   puppetSprite 39, False
   puppetSprite 40, False
   puppetSprite 41, False
   puppetSprite 42, False

  end if

  updatestage

  startTimer
  repeat while the timer < 15
   nothing
  end repeat
```

```
  puppetSound 0

 end mouseDown
```

Script 125 Digital Waypoint Toggle

```
 Global gDigWayPt

 on mouseDown

  puppetSound "click"

  if gDigWayPt = True then
   set gDigWayPt = False

   puppetSprite 44, True

   set the memberNum of Sprite 44 to 731

  else

   set gDigWayPt = True

   puppetSprite 44, False

  end if

  updatestage

  startTimer
  repeat while the timer < 15
   nothing
  end repeat

  puppetSound 0

 end mouseDown
```

Script 126 Time To Go Toggle

```
 Global gTimeToGo

 on mouseDown

  puppetSound "click"
```

```
if gTimeToGo = True then
  set gTimeToGo = False

  puppetSprite 46, True
  puppetSprite 47, True
  puppetSprite 48, True


  set the memberNum of Sprite 46 to 716
  set the memberNum of Sprite 47 to 712
  set the memberNum of Sprite 48 to 717


 else

  set gTimeToGo = True

  puppetSprite 46, False
  puppetSprite 47, False
  puppetSprite 48, False

 end if

 updatestage

 startTimer
 repeat while the timer < 15
  nothing
 end repeat

 puppetSound 0

end mouseDown
```

Script 152 Waypoint Toggle

```
--SEE WAYPOINT TOGGLE, THE WAYPOINT TURNS OFF WITH THE
COMPASS!!!


Global gCompass, gWayPt

on mouseDown  --When user clicks on the Compass or Lubber Line
```

```
puppetSound "click"  --Make a clicking noise

if   gCompass = True then --If Compass is ON

  set gCompass = False    --Then turn Compass, Lubber line
  set gWayPt   = False    --and Waypoint OFF

  puppetSprite 17, True   --Take control of the Compass
  puppetSprite 18, True   --Lubber line
  puppetSprite 19, True   --and Waypoint Sprites

  set the memberNum of Sprite 17 to 709  --Make the Compass
  set the memberNum of Sprite 18 to 691  --Lubber line
  set the memberNum of Sprite 19 to 710  --and Waypoint WHITE

else              --If the Compass was already turned OFF

  set gCompass = True    --Turn the Compass, Lubber Line
  set gWayPt  =  True    --and Waypoint back ON

  puppetSprite 17, False  --Return control of the Compass
  puppetSprite 18, False  --Lubber line
  puppetSprite 19, False  --and Waypoint Sprites to the movie


end if

updatestage    --Re-draw the screen to show status of symbol

startTimer
repeat while the timer < 15  --Delay to allow the click sound to play
  nothing
end repeat

puppetSound 0              --Turn off the click sound

end mouseDown
```

B. Lingo Script for HUDmain (main.exe)

Script 41 Hover Mountain Ridge

```
--Plays the mountain ridge hover movie

on MouseDown
```

```
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 17 to 23
  puppetSprite 17, False
  set the memberNum of Sprite 18 to 43
  puppetSprite 18, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Mountain Hover"
End
```

Script 42 Hover Desert

```
--Plays the desert hover movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 19 to 25
  puppetSprite 19, False
  set the memberNum of Sprite 20 to 43
  puppetSprite 20, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Desert Hover"
end
```

Script 44 Desert Medium non-interactive

```
--Plays the Desert Medium non-interactive movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 12 to 26
  puppetSprite 12, False
  set the memberNum of Sprite 13 to 43
```

```
  puppetSprite 13, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Medium Desert"
end
```

Script 45 Desert Shadow noninteractive

```
--Plays the noninteractive Desert Shadow Movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 10 to 27
  puppetSprite 10, False
  set the memberNum of Sprite 11 to 43
  puppetSprite 11, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Desert Shadow"
End
```

Script 57 Frontpage Loop

```
--Opening frame loop and global variable declaration


Global Rollover_sound

on exitFrame
  set Rollover_sound = False
  go loop
end
```

Script 58 Contents Page Loop

```
-- repeats the Contents frame
```

```
on exitFrame
  go to "Contents"
end
```

Script 59 Introduction Page Loop

```
-- repeats the Contents frame

on exitFrame
  go to "Contents"
end
```

Script 60 Contents Page controls

```
--Controls the Contents display user interface

on exitFrame

  --- A boolean flag to control the rollover sound

  Global Rollover_sound


  --- User interface controls and masks the cursor during rollover

  case rollover() of

    7: puppetSprite 7, True
      set the memberNum of Sprite 7 to 40
      if Rollover_sound = False then
        puppetSound "Rollover"
        set Rollover_sound = True
      end if
      cursor 200

    8: puppetSprite 8, True
      set the memberNum of Sprite 8 to 49
      if Rollover_sound = False then
        puppetSound "Rollover"
        set Rollover_sound = True
      end if
      cursor 200
```

```
9: puppetSprite 9, True
 set the memberNum of Sprite 9 to 34
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

10: puppetSprite 10, True
 set the memberNum of Sprite 10 to 39
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

11: puppetSprite 11, True
 set the memberNum of Sprite 11 to 51
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

12: puppetSprite 12, True
 set the memberNum of Sprite 12 to 38
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

13: puppetSprite 13, True
 set the memberNum of Sprite 13 to 53
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

14: puppetSprite 14, True
 set the memberNum of Sprite 14 to 37
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
```

```
 end if
 cursor 200

15: puppetSprite 15, True
 set the memberNum of Sprite 15 to 48
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

16: puppetSprite 16, True
 set the memberNum of Sprite 16 to 33
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

17: puppetSprite 17, True
 set the memberNum of Sprite 17 to 36
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

18: puppetSprite 18, True
 set the memberNum of Sprite 18 to 50
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

19: puppetSprite 19, True
 set the memberNum of Sprite 19 to 35
 if Rollover_sound = False then
  puppetSound "Rollover"
  set Rollover_sound = True
 end if
 cursor 200

20: puppetSprite 20, True
 set the memberNum of Sprite 20 to 52
```

```
  if Rollover_sound = False then
    puppetSound "Rollover"
    set Rollover_sound = True
  end if
  cursor 200

21: puppetSprite 21, True
  set the memberNum of Sprite 21 to 32
  if Rollover_sound = False then
    puppetSound "Rollover"
    set Rollover_sound = True
  end if
  cursor 200

22: puppetSprite 22, True
  set the memberNum of Sprite 22 to 31
  if Rollover_sound = False then
    puppetSound "Rollover"
    set Rollover_sound = True
  end if
  cursor 200

23: puppetSprite 23, True
  set the memberNum of Sprite 23 to 47
  if Rollover_sound = False then
    puppetSound "Rollover"
    set Rollover_sound = True
  end if
  cursor 200

24: puppetSprite 24, True
  set the memberNum of Sprite 24 to 31
  if Rollover_sound = False then
    puppetSound "Rollover"
    set Rollover_sound = True
  end if
  cursor 200

        ---Resets the initial display and unmasks the cursor

otherwise

  set the memberNum of Sprite 7 to 25
  puppetSprite 7, False
```

```
set the memberNum of Sprite 8 to 43
puppetSprite 8, False

set the memberNum of Sprite 9 to 20
puppetSprite 9, False

set the memberNum of Sprite 10 to 27
puppetSprite 10, False

set the memberNum of Sprite 11 to 43
puppetSprite 11, False

set the memberNum of Sprite 12 to 26
puppetSprite 12, False

set the memberNum of Sprite 13 to 43
puppetSprite 13, False

set the memberNum of Sprite 14 to 25
puppetSprite 14, False

set the memberNum of Sprite 15 to 43
puppetSprite 15, False

set the memberNum of Sprite 16 to 21
puppetSprite 16, False

set the memberNum of Sprite 17 to 23
puppetSprite 17, False

set the memberNum of Sprite 18 to 43
puppetSprite 18, False

set the memberNum of Sprite 19 to 25
puppetSprite 19, False

set the memberNum of Sprite 20 to 43
puppetSprite 20, False

set the memberNum of Sprite 21 to 22
puppetSprite 21, False

set the memberNum of Sprite 22 to 24
puppetSprite 22, False
```

```
    set the memberNum of Sprite 23 to 43
    puppetSprite 23, False

    set the memberNum of Sprite 24 to 19
    puppetSprite 24, False

    cursor -1
    set Rollover_sound = False
    puppetSound 0
  end case

  updatestage
  go to "Contents"
end
```

Script 61 Exit Page

```
--End of program control

on exitFrame
  startTimer
  repeat while the timer < 60
    nothing
  end repeat
  quit
end
```

Script 62 Introduction button

```
--Introduction button control

on exitframe
  if rollOver (4) then
    puppetSprite 4, True
    set the memberNum of Sprite 4 to 10
  else
    set the memberNum of Sprite 4 to 9
    puppetSprite 4, False
  end if
  updatestage
end

on mouseDown
  puppetSprite 4, True
  set the memberNum of Sprite 4 to 11
```

```
  puppetSound "click"
  updatestage
  go to "Introduction"
  startTimer
  repeat while the timer < 30
    nothing
  end repeat
end mouseDown

on mouseUp
  set the memberNum of Sprite 4 to 9
  updatestage
  startTimer
  repeat while the timer < 30
    nothing
  end repeat
  puppetSprite 4, False
  puppetSound 0

end mouseUp
```

Script 63 Contents Button

```
--Contents button control

on exitframe
  if rollOver (5) then
    puppetSprite 5, True
    set the memberNum of Sprite 5 to 13
  else
    set the memberNum of Sprite 5 to 12
    puppetSprite 5, False
  end if
  updatestage
end

on mouseDown
  puppetSprite 5, True
  set the memberNum of Sprite 5 to 14
  puppetSound "click"
  updatestage
  go to "Contents"
  startTimer
  repeat while the timer < 30
    nothing
```

```
  end repeat

end mouseDown

on mouseUp
  set the memberNum of Sprite 5 to 12
  updatestage
  startTimer
  repeat while the timer < 30
    nothing
  end repeat
  puppetSprite 5, False
  puppetSound 0

end mouseUp
```

Script 64 Exit Button

```
--Exit Button control

on exitframe
  if rollOver (6) then
    puppetSprite 6, True
    set the memberNum of Sprite 6 to 16
  else
    set the memberNum of Sprite 6 to 15
    puppetSprite 6, False
  end if
  updatestage
end

on mouseDown
  puppetSprite 6, True
  set the memberNum of Sprite 6 to 17
  puppetSound "click"
  updatestage
  go to "Exit"
  startTimer
  repeat while the timer < 30
    nothing
  end repeat
end mouseDown

on mouseUp
  set the memberNum of Sprite 6 to 15
```

```
  updatestage
  startTimer
  repeat while the timer < 30
    nothing
  end repeat
  puppetSprite 6, False
  puppetSound 0

end mouseUp
```

Script 65 Hud Tutorial Control

```
--Plays the hud tutorial movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 22 to 24
  puppetSprite 22, False
  set the memberNum of Sprite 23 to 43
  puppetSprite 23, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Demo"
End
```

Script 66 Desert Bright Interactive

```
--Plays the interactive Desert Bright movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 7 to 25
  puppetSprite 7, False
  set the memberNum of Sprite 8 to 43
  puppetSprite 8, False
  puppetsound 0
  cursor -1
```

```
  go to "Movie Jump"
  Play movie "Interactive"
End
```

Script 67 Desert Bright Noninteractive

```
--Plays the noninteractive Desert Bright Movie

on MouseDown
  puppetsound "click"
end

on mouseUp
  set the memberNum of Sprite 14 to 25
  puppetSprite 14, False
  set the memberNum of Sprite 15 to 43
  puppetSprite 15, False
  puppetsound 0
  cursor -1
  go to "Movie Jump"
  Play movie "Bright Desert"
end
```

APPENDIX C. INSTRUCTIONS FOR SETUP OF DEMONSTRATION SYSTEM

A. SYSTEM REQUIREMENTS

Recommended:	Computer	Pentium Pro II 266 MHz
		128 Mbytes RAM
		1.5 Gbytes of available Hard Disk space
		Sound Card
		External Speakers
		Microphone
		Forte VFX1 Headgear HMD
		Hercules Dynamite™ 128 Video Card
	Software	Windows 95
		Demonstration Projector (executable file)
		Demonstration modules (listed below)
		DirectControl Xtras for viewing

The demonstration can be run on slower systems with less memory without an HMD. The HMD provides for a more life-like medium for viewing the imagery. A slower system will likely run at a frame rate significantly slower than intended. Significant lag in panning the scenes will occur on slower systems. Using the recommended system allows playback to occur at higher and more desirable playback rates thus providing a higher refresh rate and minimizing update lag.

B. SET UP AND CONFIGURATION FOR THE RECOMMENDED SYSTEM

1. Ensure that the most current drivers and bios are being used for all installed hardware. Contact the manufacturer for verification. These will deliver the best performance for your system.
2. Remove the current video card and install the Hercules Dynamite™ 128 video card. (For a list of video cards compatible with the VFX1 contact Forte).
3. Install the VFX1 VIP card in accordance with the manufacturer's instructions. Attach the HMD to the computer as per the instructions. All setup and calibration instructions should be followed. The VFX1 joystick driver should be installed on joystick number one and the puck should be installed as joystick number two and calibrate.
4. Copy the demonstration projector, main.exe and associate files into a single directory.

Name	Description
Main.exe	Main menu/startup screen
Tutor.dir	HUD tutorial movie
Bright Desert.dir	Flight over highly illuminated desert terrain
Medium Desert.dir	Flight over moderately illuminated desert terrain
Desert Shadow.dir	Flight over desert terrain with low angle moon
Desert Hover.dir	Hover over desert terrain, allows HMD pan
Mountain Hover.dir	Hover over mountainous terrain, allows HMD pan

Interactive.dir	Flight over desert terrain, allows HMD pan
Xtras	A directory containing the Direct Control Xtra file names, D Ctrl_R.x32 and D Ctrl_R.x16

5. Change the resolution of the screen to 640 X 480, 256 color and 60 Hz refresh rate.
6. Disable all background jobs including virus detectors etc.
7. If an instructor or external observer will be present it is useful to turn on the system microphone and adjust the volume so he can communicate with the HMD user. The sound track that is part of the multimedia experience obscures external noises for the HMD user.
8. Open the **VFX1 calibration** program and ensure that none of the options are checked on. At this point the user should don the HMD and face in the direction he would like to be forward. Care should be exercised to ensure that the area around the user is free of obstruction and that the user is seated for safety. The safety notes from the VFX1 instruction manual should be followed. Click on Set Zero to establish this position as the null (forward) position. Take a few minutes to focus the goggles at this point. After the goggles are focused flip them up until instructed by the VESD program to put them down.
9. Run the demonstration projector (main.exe) file.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudly Knox Library 2
   Naval Postgraduate School
   411 Dryer Rd.
   Monterey, CA 93943-5101

3. Tung Bui, Code SM/SE 1
   Department of Systems Management
   Naval Postgraduate School
   Monterey, CA 93943-5002

4. Anthony Ciavarelli, Code 10 2
   Aviation Safety Programs
   Naval Postgraduate School
   Monterey, CA 93942-5002

5. Commander Naval Air Systems Command 2
   (PMA-205) Training Systems
   1421 Jefferson Highway (JP-1, Rm. 306)
   Arlington, VA 22243-1205

6. LT G. Thomas Foggin IV 1
   710 Eigth St. N.E.
   Washington D.C. 20002-3604

7. LT Paul J. O'Rourke 1
   251 Windmill Road
   West Seneca, NY 14218

8. Joseph C. Antonio, M.D. 1
   Human Resources Directorate
   Aircrew Training Research Division
   6001 So. Power Road, Bldg. 558
   Mesa, AZ 85206–0904

9. Marine Aviation Weapons and Tactics Squadron One 1
   Box 99200
   Yuma, AZ 85369–9200

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

DUDLEY KNOX LIBRARY
3 2768 00338990 9